

Risk Management Framework (RMF)

Gary E. McGraw, Cigital, Inc. [vita³]

Copyright © 2005 Cigital, Inc.

2005-09-21

L4 / L, M⁴

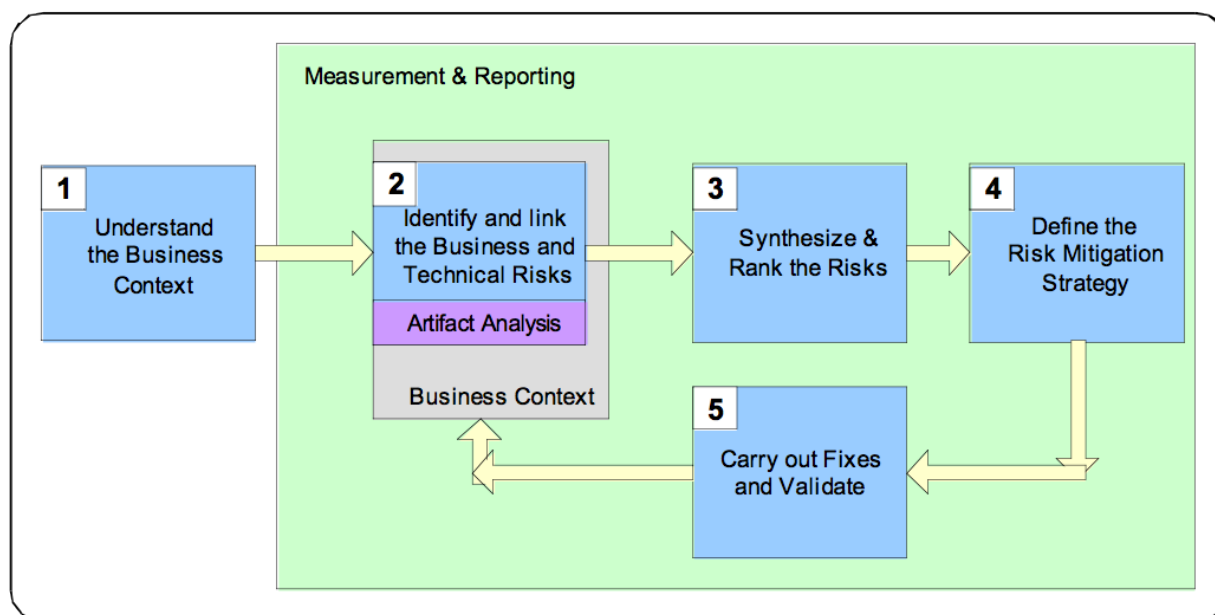
A continuous risk management process is a necessary part of any approach to software security. Software security risk includes risks found in artifacts during assurance activities, risks introduced by insufficient process, and personnel related risks. An overall risk management framework (described here) can help make sense of software security. Note that we are explicitly teasing apart architectural risk analysis (one of the critical software security best practices) and use of the risk management framework.

A risk management framework is an essential philosophy for approaching security work. Following the risk management framework introduced here is by definition a full life-cycle activity. For the purposes of this description, consider risk management a high-level approach to iterative risk analysis that is deeply integrated throughout the software development life cycle (SDLC).

The RMF described here is a condensed version of the Cigital RMF, a mature process that has been applied in the field for almost ten years. This RMF is designed to manage software-induced business risks. Through the application of five simple activities, analysts use their own technical expertise, relevant tools, and technologies to carry out a reasonable risk management approach.

The purpose of an RMF like this is to allow a consistent and repeatable expertise-driven approach to risk management. As we converge on and describe software risk management activities in a consistent manner, the basis for measurement and common metrics emerges. Such metrics are sorely needed and should allow organizations to better manage business and technical risks given particular quality goals; make more informed, objective business decisions regarding software (e.g., whether an application is ready to release); and improve internal software development processes so that they in turn better manage software risks.

Figure 1. The BSI risk management framework



3. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/198-BSI.html (McGraw, Gary)

Figure 1 shows the RMF as a closed loop process with five basic activity stages. Throughout the application of the RMF, measurement and reporting activities occur. These activities focus on tracking, displaying, and understanding progress regarding software risk.

Five Stages of Activity

The RMF consists of the five fundamental activity stages shown in Figure 1:

1. Understand the business context.
2. Identify the business and technical risks.
3. Synthesize and prioritize the risks, producing a ranked set.
4. Define the risk mitigation strategy.
5. Carry out required fixes and validate that they are correct.

Each of the stages is briefly summarized here. Critical business decisions, including release readiness, can be made in a more straightforward and informed manner by identifying, tracking, and managing software risk explicitly as described in the RMF.

1. Understand the Business Context

Software risk management occurs in a business context. Risks are unavoidable and are a necessary part of software development. Management of risks, including the notions of risk aversion and technical tradeoff, is deeply impacted by business motivation. Thus, the first stage of software risk management involves getting a handle on the business situation. Commonly, business goals are neither obvious nor explicitly stated. In some cases, you may even have difficulty expressing these goals clearly and consistently. During this stage, the analyst must extract and describe business goals, priorities, and circumstances in order to understand what kinds of software risks to care about and which business goals are paramount. Business goals include, but are not limited to, increasing revenue, meeting service level agreements, reducing development costs, and generating high return on investment.

2. Identify Business and Technical Risks

Business risks directly threaten one or more of a customer's business goals. The identification of such risks helps to clarify and quantify the possibility that certain events will directly impact business goals. Business risks have impacts that include direct financial loss, damage to brand or reputation, violation of customer or regulatory constraints, exposure to liability, and increase in development costs. The severity of a business risk should be expressed in terms of financial or project management metrics. These include, but are not limited to, market share (%), direct cost, level of productivity, and cost of rework.

Business risk identification helps to define and steer use of particular technical methods for extracting, measuring, and mitigating software risk given various software artifacts. The identification of business risks provides a necessary foundation that allows software risk (especially impact) to be quantified and described in business terms.

The key to making risk management work for business lies in tying technical risks to business context in a meaningful way. The ability to identify and deeply understand risks is thus essential. Uncovering and recognizing technical risks is a high-expertise undertaking that usually requires years of experience.

Central to this stage of the RMF is the ability to discover and describe technical risks and map them (through business risks) to business goals. A technical risk is a situation that runs counter to the planned design or implementation of the system under consideration. For example, a technical risk may give rise to the system behaving in an unexpected way, violating its own design strictures, or failing to perform as required. Technical risks can also be related to the process of building software. The process an organization follows may offer too many opportunities for mistakes in design or implementation. Technical risks involve impacts such as unexpected system crashes, avoidance of controls (audit or otherwise), unauthorized data modification or disclosure, and needless rework of artifacts during development.

Technical risk identification is supported by the software security best practices described on this portal, especially those best practices that involve artifact analysis.

3. Synthesize and Prioritize Risks

Large numbers of risks will be apparent in almost any given system. Identifying these risks is important, but it is the prioritization of these risks that leads directly to creation of value. Through the activities of synthesizing and prioritizing risks, the critical "Who cares?" question can (and must) be answered. Synthesis and prioritization should be driven to answer questions such as "What shall we do first given the current risk situation?" and "What is the best allocation of resources, especially in terms of risk mitigation activities?" Clearly, the prioritization process must take into account which business goals are the most important to the organization, which goals are immediately threatened, and how likely technical risks are to manifest themselves in such a way as to impact the business. This stage creates as its output a list of all the risks and their appropriate priority for resolution. Typical risk metrics include, but are not limited to, risk likelihood, risk impact, risk severity, and number of risks emerging and mitigated over time.

4. Define the Risk Mitigation Strategy

Given a set of risks and their priorities from stage three, the next stage is to create a coherent strategy for mitigating the risks in a cost effective manner. Any suggested mitigation activities must take into account cost, time to implement, likelihood of success, completeness, and impact over the entire corpus of risks. A risk mitigation strategy must be constrained by the business context and should consider what the organization can afford, integrate, and understand. The strategy must also directly identify validation techniques that can be used to demonstrate that risks are properly mitigated. Typical metrics to consider in this stage are financial in nature and include estimated cost takeout, return on investment, method effectiveness in terms of dollar impact, and percentage of risk coverage (related in terms of removing costly impact).

5. Fix the Problems and Validate the Fixes

Once a mitigation strategy has been defined, it must be executed. Those artifacts where problems (e.g., architectural flaws in a design, requirements collisions, or problems in testing) have been identified should be rectified. Risk mitigation is carried out according to the strategy defined in stage four. Progress at this stage should be measured in terms of completeness against the risk mitigation strategy. Good metrics include, but are not limited to, progress against risks, open risks remaining, and any artifact quality metrics previously identified.

This stage also involves application of those validation techniques previously identified. The validation stage provides some confidence that risks have been properly mitigated through artifact improvement and that the risk mitigation strategy is working. Testing can be used to demonstrate and measure the effectiveness of risk mitigation activities. The central concern at this stage is to validate that software artifacts and processes no longer bear unacceptable risks. This stage should define and leave in place a repeatable, measurable, verifiable validation process that can be run from time to time to continually verify artifact quality. Typical metrics employed during this stage include artifact quality metrics as well as levels of risk mitigation effectiveness.

Measurement and Reporting on Risk

The activity of identifying, tracking, storing, measuring, and reporting software risk information cannot be overemphasized. Successful use of the RMF depends on continuous and consistent identification and storage of risk information as it changes over time. A master list of risks should be maintained during all stages of RMF execution and continually revisited. Measurements regarding this master list make excellent reporting information. For example, the number of risks identified in various software artifacts and/or software life-cycle phases can be used to identify problematic areas in the software process. Likewise, the number of risks mitigated over time can be used to show concrete progress as risk mitigation activities unfold.

The RMF is a Multilevel Loop

The RMF shown in Figure 1²⁹ has a clear loop that represents the idea that risk management is a continuous process. That is, identifying risks only once during a software project is insufficient. The idea of "crossing off" a particular stage once it has been executed and never doing those activities again is incorrect. Though the five stages are shown in a particular serial order in Figure 1³⁰, they may need to be applied over and over again throughout a project, and their particular ordering may be interleaved in many different ways.

There are two main reasons for this complication. First, risks can crop up at any time during the software life cycle. One natural way to apply a cycle of the loop is during each particular software life-cycle phase. For example, software risks should be identified, ranked, and mitigated (one loop) during requirements and again during design (another loop). Secondly, risks can crop up between stages, regardless of where in the process a project finds itself.

In addition to the issue of continuous looping is a further complication regarding level of application. Put simply, the RMF is fractal; that is, the entire process can be applied at several different levels. The primary level is the project level. Each stage of the loop clearly must have some representation during a complete engagement in order for risk management to be effective. Another level is the software life-cycle phase level. The loop will most likely have a representation at the requirements phase, the design phase, the architecture phase, the test planning phase, and so on. A third level is the artifact level. The loop will have a representation during both requirements analysis and use case analysis, for example. Fortunately, a generic description of the validation loop as a serial looping process is sufficient to capture critical aspects at all of these levels at once.

In order to facilitate the learning process, this document presents the RMF as a series of stages, tasks, and methods that can be performed in succession, each stage following a particular process and producing a new set of work products and metrics that enhance and clarify previously created data sets. We describe how the RMF encompasses a particular cycle of the loop that is repeatedly executed on more than one level. The RMF loop restarts continuously so that newly arising business and technical risks can be identified and the status of existing risks currently undergoing mitigation can be kept up.

Understanding that the risk management process is by nature cumulative and at times arbitrary and difficult to predict (depending on project circumstances) is an important insight. Given this insight, we acknowledge that the practice of specific RMF stages, tasks, and methods (as described serially here for pedagogical reasons) may occur independently of one another, in parallel, repeatedly, and unsystematically.

Analysts may "skip through" an analytical process, as information gained from the performance of one activity may require the analyst to perform an activity located earlier, or several steps later, in the process cycle. For instance, after finding a rare technical risk, an analyst may need to conduct additional research prior to reprioritizing the risk tables and updating the risk mitigation strategy. In light of our discussion, users of this process should focus more on the basic concepts and activities presented here than on the serial order they are presented in.

In practice, less experienced analysts should rely on following our processes as closely as possible, preserving ordering and proceeding in continuous loops. Expert analysts are likely to devise work patterns that use the concepts and processes described here in a less ordered manner.

Business case

Central to the notion of risk management is the idea of clearly describing impact. Without a clear and compelling tie to either business or mission consequences, technical risks, software defects, and the like are not often compelling enough on their own to spur action. The risks we focus on in this portal are all tied directly to software and all have clear security ramifications. However, unless these risks are described in terms that business people and decision makers understand, they will not likely be addressed. The ever-

29. #dsy250-BSI_figure-1

30. #dsy250-BSI_figure-1

increasing integration of business processes and IT systems means that software risks can often be linked to serious and specific impacts on the mission of an organization or business. Since resources are rarely unlimited, mitigation of software risks can and should be prioritized according to the severity of the related business risks.

Hence, software risk management can only be successfully carried out in a business context. Risks are unavoidable and are a necessary part of software development. Management of risks, including the notion of risk aversion and technical tradeoff, is deeply impacted by business motivation. Thus, the first stage of software risk management involves assessing the business situation. Commonly, business goals are neither obvious nor explicitly stated. In some cases, you may even have difficulty expressing these goals clearly and consistently. During this stage, the analyst must extract and describe business goals, priorities, and circumstances in order to understand what kinds of software risks to care about and which business goals are paramount. Business goals include, but are not limited to, increasing revenue, meeting service level agreements, reducing development costs, and generating high return on investment.

The activities of identifying, tracking, storing, measuring, and reporting software risk information cannot be overemphasized. Successful use of the RMF depends on continuous and consistent identification and storage of risk information as it changes over time. A master list of risks should be maintained during all stages of RMF execution and continually revisited. Measurements regarding this master list make excellent reporting information. For example, the number of risks identified in various software artifacts and/or software life-cycle phases can be used to identify problematic areas in software process. Likewise, the number of software risks mitigated over time can be used to show concrete progress as risk mitigation activities unfold. Links to descriptions or measurements of the corresponding business risks mitigated can be used to clearly demonstrate the business value of the software risk mitigation process and the risk management framework.

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>